


```
LL      000000  EEEEEEEEE  MM      MM  P P P P P P P  HH      HH
LL      000000  EEEEEEEEE  MM      MM  P P P P P P P  HH      HH
LL      00      00  EE          MMMM  MMMM  PP      PP  HH      HH
LL      00      00  EE          MMMM  MMMM  PP      PP  HH      HH
LL      00      00  EE          MM  MM  MM  PP      PP  HH      HH
LL      00      00  EE          MM  MM  MM  PP      PP  HH      HH
LL      00      00  EEEEEEEEE  MM      MM  P P P P P P P  HHHHHHHHHH
LL      00      00  EEEEEEEEE  MM      MM  P P P P P P P  HHHHHHHHHH
LL      00      00  EE          MM      MM  PP      PP  HH      HH
LL      00      00  EE          MM      MM  PP      PP  HH      HH
LL      00      00  EE          MM      MM  PP      PP  HH      HH
LL      00      00  EE          MM      MM  PP      PP  HH      HH
LL      00      00  EE          MM      MM  PP      PP  HH      HH
LL      00      00  EEEEEEEEE  MM      MM  PP      PP  HH      HH
LLLLLLLL 000000  EEEEEEEEE  MM      MM  PP      PP  HH      HH
LLLLLLLL 000000  EEEEEEEEE  MM      MM  PP      PP  HH      HH
```

```
LL      I I I I I  S S S S S S S
LL      I I I I I  S S S S S S S
LL      I I      S S
LL      I I      S S
LL      I I      S S
LL      I I      S S
LL      I I      S S S S S
LL      I I      S S S S S
LL      I I      S S
LL      I I      S S
LL      I I      S S
LL      I I      S S
LL      I I      S S
LL      I I      S S
LLLLLLLL I I I I I  S S S S S S S
LLLLLLLL I I I I I  S S S S S S S
```

.....


```
0001 0 %TITLE 'Line output (emphasis -- bolding and underlining); overstriking'
0002 0 MODULE LOEMPH (
0003 0 IDENT = 'V04-000'
P 0004 0 %BLISS32C,
P 0005 0 ADDRESSING_MODE(EXTERNAL=LONG_RELATIVE, NONEXTERNAL=LONG_RELATIVE)
0006 0 ]
0007 0 ) =
0008 1 BEGIN
0009 1
0010 1 *****
0011 1 *
0012 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0013 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0014 1 * ALL RIGHTS RESERVED.
0015 1 *
0016 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0017 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0018 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0019 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0020 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0021 1 * TRANSFERRED.
0022 1 *
0023 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0024 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0025 1 * CORPORATION.
0026 1 *
0027 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0028 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0029 1 *
0030 1 *
0031 1 *****
0032 1
0033 1 ++
0034 1 FACILITY: DSR (Digital Standard RUNOFF) / DSRPLUS
0035 1
0036 1 ABSTRACT: Translation from intermediate format to final output.
0037 1
0038 1
0039 1 ENVIRONMENT: Transportable
0040 1
0041 1 AUTHOR: K. A. Dawson CREATION DATE: December 1983
0042 1
```


LOEMPH
V04-000

Line output (emphasis -- bolding and underlinin K 1
Revision History 16-Sep-1984 00:49:27
14-Sep-1984 13:06:55

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]LOEMPH.BLI;1

Page 2
(2)

: 44 0043 1 %SBTTL 'Revision History'
: 45 0044 1
: 46 0045 1 MODIFIED BY:
: 47 0046 1
: 48 0047 1 001 KAD00001
: 49 0048 1
: 50 0049 1 !--

Keith Dawson 22-Mar-1983


```
52 0050 1 %SBTTL 'Module Level Declarations'
53 0051 1
54 0052 1
55 0053 1 !! TABLE OF CONTENTS:
56 0054 1 !!
57 0055 1
58 0056 1 REQUIRE 'REQ:RNODEF';                ! RUNOFF variant definitions
59 0187 1
60 0188 1 FORWARD ROUTINE
61 0189 1     bsemph : NOVALUE,
62 0190 1     opemph  : NOVALUE
63 0191 1 %IF LN01 %THEN
64 0192 1     lnemph : NOVALUE
65 0193 1 %FI
66 U 0194 1 %IF DSRPLUS %THEN
67 U 0195 1     vtemph : NOVALUE
68 0196 1 %FI
69 U 0197 1 %IF FLIP %THEN
70 U 0198 1     flemp  : NOVALUE
71 0199 1 %FI
72 0200 1 ;
73 0201 1
74 0202 1
75 0203 1 !! INCLUDE FILES:
76 0204 1 !!
77 0205 1 LIBRARY 'NXPORT:XPORT';                ! XPORT Library
78 0206 1
79 U 0207 1 %IF DSRPLUS %THEN
80 U 0208 1 LIBRARY 'REQ:DPLLIB';                ! DSRPLUS BLISS Library
81 0209 1 %ELSE
82 0210 1 LIBRARY 'REQ:DSRLIB';                ! DSR BLISS Library
83 0211 1 %FI
84 0212 1
85 0213 1 !! MACROS:
86 0214 1 !!
87 0215 1 MACRO
88 M 0216 1     write emphasis =
89 M 0217 1     BEGIN
90 M 0218 1     LOCAL
91 M 0219 1     ptr;
92 M 0220 1     ptr = .work_string[STR$A POINTER];
93 M 0221 1     INCR i FROM 1 TO .work_string[STR$H_LENGTH] DO
94 M 0222 1         fs_wchar (fra, CHRCHAR_A(ptr) );
95 M 0223 1     END
96 0224 1     %;
97 0225 1
98 0226 1
99 0227 1 !! EQUATED SYMBOLS:
100 0228 1 !!
101 0229 1
102 0230 1 EXTERNAL LITERAL
103 0231 1     rintex : UNSIGNED (8);
104 0232 1
105 0233 1 LITERAL
106 0234 1     max_output_line_length = 250,        ! Reasonable maximum length for an output line.
107 0235 1     backspace = %0'T0',
108 0236 1     escape = %0'33';
```


LOEMPH
V04-000

Line output (emphasis -- bolding and underlinin
Module Level Declarations

M 1
16-Sep-1984 00:49:27
14-Sep-1984 13:06:55

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]LOEMPH.BLI;1

Page 4
(3)

```
: 109      0237 1
: 110      0238 1
: 111      0239 1  | OWN STORAGE:
: 112      0240 1
: 113      0241 1  | OWN
: 114      0242 1      work_string : $STR_DESCRIPTOR ( CLASS=DYNAMIC, STRING=(0,0) );
: 115      0243 1
: 116      0244 1
: 117      0245 1  | EXTERNAL REFERENCES:
: 118      0246 1
: 119      0247 1
: 120      0248 1  | EXTERNAL
: 121      0249 1      fra : fixed_string,
: 122      0250 1      outopt : VECTOR [outopt_size],
: 123      0251 1      tsf : tsf_definition;
: 124      0252 1
: 125      0253 1  | EXTERNAL ROUTINE
: 126      0254 1      clh,          erms;
```



```
128 0255 1 %SBTTL 'BSEMPH -- do emphasis by backspacing'
129 0256 1 GLOBAL ROUTINE BSEMPH
130 0257 1 ( character
131 0258 1 , italics
132 0259 1 , adr_emphasis_bits
133 0260 1 , overstrike_count
134 0261 1 , overstrike_char
135 0262 1 , overstrike_seq
136 0263 1 , pass_cntr
137 0264 1 ) : NOVALUE =
138 0265 1
139 0266 1 ++
140 0267 1 FUNCTIONAL DESCRIPTION:
141 0268 1
142 0269 1 BSEMPH handles emphasis -- bolding and underlining -- if the
143 0270 1 user said /BACKSPACE on the command line.
144 0271 1
145 0272 1 FORMAL PARAMETERS:
146 0273 1
147 0274 1 character is the current character to be output (emphasized).
148 0275 1 italics Not used by this routine -- passed for conformance only.
149 0276 1 adr_emphasis_bits Address of a word containing information on current-
150 0277 1 character and previous-character bold and underline.
151 0278 1 overstrike_count Number of characters in an overstrike sequence.
152 0279 1 overstrike_char The character with which to overstrike the previous one.
153 0280 1 overstrike_seq CH$PTR to the start of an overstrike sequence.
154 0281 1 pass_cntr Count of which pass is happening.
155 0282 1
156 0283 1 IMPLICIT INPUTS: None
157 0284 1
158 0285 1 IMPLICIT OUTPUTS: None
159 0286 1
160 0287 1 ROUTINE VALUE:
161 0288 1 COMPLETION CODES: None
162 0289 1
163 0290 1 SIDE EFFECTS: None
164 0291 1 --
165 0292 1
166 0293 2 BEGIN
167 0294 2
168 0295 2 BIND emphasis_bits = .adr_emphasis_bits;
169 0296 2
170 0297 2 LOCAL
171 0298 2 count;
172 0299 2
173 0300 2 ! Before processing the character, make sure that (1) there will be sufficient space in the
174 0301 2 ! output buffer, and (2) the output line is not too long for the operating system to handle.
175 0302 2 IF .fs_length (fra) GTR (.fs_maxsize (fra) - 20) ! Need at least 20 characters.
176 0303 2 OR
177 0304 2 .fs_length (fra) GTR max_output_line_length ! Reasonable output-line maximum length
178 0305 2 THEN
179 0306 2 BEGIN
180 0307 2 clh (clh_out_nocrlf); ! Output what's been built up
181 0308 2 fs_init (fra); ! so far and start filling a
182 0309 2 END; ! new buffer
183 0310 2
184 0311 2 IF .emph_current_bold
```



```
185 0312 2 THEN
186 0313 2 ! Protective code against /BOLD:<ridiculous amount> and
187 0314 2 count = MIN (10, .outopt_bldn + 1) ! stops buffer overflow.
188 0315 2 ELSE
189 0316 2 count = 1;
190 0317 2
191 0318 2 ! Repeatedly process the character as many times as it is
192 0319 2 ! to output. In most cases, this is once. But if the
193 0320 2 ! character is bolded it will be scanned several times.
194 0321 2 INCR i FROM 1 TO .count DO
195 0322 2 BEGIN
196 0323 2 ! If bolding, output a backspace before each re-scan
197 0324 2 ! of the character.
198 0325 2 IF .i GTR 1
199 0326 2 THEN
200 0327 2 fs_wchar (fra, backspace);
201 0328 2
202 0329 2 ! Look for underlining
203 0330 2 IF .emph_current_underline
204 0331 2 THEN
205 0332 2 BEGIN
206 0333 2 fs_wchar (fra, .outopt_und_char);
207 0334 2
208 0335 2 ! Don't backspace if underscore is non-spacing.
209 0336 2 IF NOT .outopt_und_nosp
210 0337 2 THEN
211 0338 2 fs_wchar (fra, backspace)
212 0339 2 END;
213 0340 2
214 0341 2 ! Output the deferred character.
215 0342 2 fs_wchar (fra, .character);
216 0343 2
217 0344 2 ! Look for overstrike.
218 0345 2 IF .overstrike_count NEQ 0
219 0346 2 THEN
220 0347 2 BEGIN
221 0348 2 LOCAL
222 0349 2 temp_seq_ptr;
223 0350 2 temp_seq_ptr = .overstrike_seq;
224 0351 2
225 0352 2 INCR i FROM 1 TO .overstrike_count DO
226 0353 2 BEGIN
227 0354 2 ! Rescan overstrike sequence to take care of
228 0355 2 ! multiple overstriking.
229 0356 2 LOCAL
230 0357 2 x;
231 0358 2
232 0359 2 x = CH$RCHAR_A (temp_seq_ptr); ! Point to the 'o';
233 0360 2 x = CH$RCHAR_A (temp_seq_ptr); ! Point to overstrike character
234 0361 2 x = CH$RCHAR_A (temp_seq_ptr); ! Get character, advance
235 0362 2 fs_wchar (fra, backspace);
236 0363 2 fs_wchar (fra, .x);
237 0364 2 END;
238 0365 2 END
239 0366 2 END
240 0367 1 END; ! End of BSEMPH
```


Line output (emphasis -- bolding and underlining)
BSEMPH -- do emphasis by backspacing

16-Sep-1984 00:49:27
14-Sep-1984 13:06:55

Page 7
(4)

.IDENT \V04-000\

.PSECT SOWNS,NOEXE,2

```
0000 00000 WORK_STRING:
```

02 0E 00002
00000000 00004

WORD	0
BYTE	14, 2
LONG	0

.EXTRN RINTES, FRA, OUTOPT
.EXTRN TSF, CLH, ERMS

.PSECT \$CODE\$,NOWRT,2

PC	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419
----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

LOEMPH
V04-000

Line output (emphasis -- bolding and underlining
BSEMPH -- do emphasis by backspacing

D 2
16-Sep-1984 00:49:27
14-Sep-1984 13:06:55

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]LOEMPH.BLI;1

Page 8
(4)

	50	18	28	13	0008E	BEQL	11\$		
			AC	D0	00090	MOVL	OVERSTRIKE_SEQ, TEMP_SEQ_PTR	:	0350
			51	D4	00094	CLRL	I	:	0352
			1B	11	00096	BRB	10\$:	
	53		80	9A	00098	MOVZBL	(TEMP_SEQ_PTR)+, X	:	0359
	53		80	9A	0009B	MOVZBL	(TEMP_SEQ_PTR)+, X	:	0360
	53		80	9A	0009E	MOVZBL	(TEMP_SEQ_PTR)+, X	:	0361
00	B5		08	90	000A1	MOVB	#8, @FRA+4	:	0362
			65	D6	000A5	INCL	FRA+4	:	
		08	A5	D6	000A7	INCL	FRA+12	:	
00	B5		53	90	000AA	MOVB	X, @FRA+4	:	0363
			65	D6	000AE	INCL	FRA+4	:	
		08	A5	D6	000B0	INCL	FRA+12	:	
E0	51	10	AC	F3	000B3	AOBLEQ	OVERSTRIKE_COUNT, I, 9\$:	0352
9B	52		54	F3	000B8	AOBLEQ	COUNT, I, 8\$:	0345
			04	000BC	RET			:	0367

; Routine Size: 189 bytes, Routine Base: \$CODE\$ + 0000


```
242 0368 1 %SBTTL 'OPEMPH -- do emphasis by overprinting'
243 0369 1 GLOBAL ROUTINE OPEMPH
244 0370 1 ( character
245 0371 1 , italics
246 0372 1 , adr_emphasis_bits
247 0373 1 , overstrike_count
248 0374 1 , overstrike_char
249 0375 1 , overstrike_seq
250 0376 1 , pass_cntr
251 0377 1 ) : NOVALUE =
252 0378 1
253 0379 1 ++
254 0380 1 FUNCTIONAL DESCRIPTION:
255 0381 1
256 0382 1 OPEMPH processes emphasis -- bolding and underlining -- for
257 0383 1 the normal case in which the user did not say either /BACK,
258 0384 1 /DEC_INTERNAL=FLIP, /DEVICE=VT100, or DEVICE=LN01[e].
259 0385 1
260 0386 1 FORMAL PARAMETERS:
261 0387 1
262 0388 1 character is the current character to be output (emphasized).
263 0389 1 italics Not used by this routine -- passed for conformance only.
264 0390 1 adr_emphasis_bits Address of a word containing information on current-
265 0391 1 character and previous-character bold and underline.
266 0392 1 overstrike_count Number of characters in an overstrike sequence
267 0393 1 (passed for conformance only).
268 0394 1 overstrike_char The character with which to overstrike the previous one.
269 0395 1 overstrike_seq CH$PTR to the start of an overstrike sequence.
270 0396 1 pass_cntr Count of which pass is happening.
271 0397 1
272 0398 1 IMPLICIT INPUTS: None
273 0399 1
274 0400 1 IMPLICIT OUTPUTS: None
275 0401 1
276 0402 1 ROUTINE VALUE:
277 0403 1 COMPLETION CODES: None
278 0404 1
279 0405 1 SIDE EFFECTS: None
280 0406 1 --
281 0407 1
282 0408 2 BEGIN
283 0409 2
284 0410 2 BIND emphasis_bits = .adr_emphasis_bits;
285 0411 2
286 0412 2 SELECT .pass_cntr OF
287 0413 2 SET
288 0414 2
289 0415 2 [pass_setup, pass_bold] :
290 0416 3 BEGIN
291 0417 3 ! Generate non-spacing underscore if requested
292 0418 3
293 0419 4 IF (.emph_current_underline
294 0420 4 AND .outopt_und_nosp
295 0421 4 AND NOT .outopt_und_sep)
296 0422 4 THEN
297 0423 3 IF (.pass_cntr EQL pass_setup) OR .emph_current_bold
298 0424 3 THEN
```



```
299      0425      3      fs_wchar (fra, .outopt_und_char);
300      0426      3
301      0427      3      ! Generate character
302      0428      4      IF (.pass_cntr EQL pass_setup) OR (.emph_current_bold)
303      0429      3      THEN
304      0430      4      fs_wchar (fra, .character)
305      0431      3      ELSE
306      0432      4      fs_wchar (fra, %C' ')
307      0433      4      END;
308      0434      2
309      0435      2      [pass_overstrike, pass_bold_overstrike] :
310      0436      2      ! Process overstriking. At this point it is only known that this
311      0437      2      character is a special character, and that overstriking is being
312      0438      2      processed. It has not yet been determined whether or not this
313      0439      2      character is to be overstruck. Just putting out overstrike_char
314      0440      2      will result in NULLs being output if this character is not to be
315      0441      2      overstruck, but is none-the-less a special character. Using MAX
316      0442      2      makes sure that NULL never gets output. This makes an implicit
317      0443      2      restriction, i.e., that the user will never try to overstrike
318      0444      2      with a character LSS %C' '. If this is unduly restrictive, MAX can
319      0445      2      be replaced with a simple test to see if overstrike_char is NULL
320      0446      2      or not.
321      0447      2
322      0448      3      IF (.pass_cntr EQL pass_overstrike) OR (.emph_current_bold)
323      0449      3      THEN
324      0450      3      fs_wchar (fra, MAX(%C' ', .overstrike_char))
325      0451      2      ELSE
326      0452      2      fs_wchar (fra, %C' ');
327      0453      2
328      0454      2      [pass_underline] :
329      0455      2      ! Process underlining.
330      0456      2      IF .emph_current_underline
331      0457      2      THEN
332      0458      3      fs_wchar (fra, .outopt_und_char)
333      0459      2      ELSE
334      0460      2      fs_wchar (fra, %C' ');
335      0461      2
336      0462      2      [pass_bold_underline] :
337      0463      2      ! Process underlining if also bold.
338      0464      2      IF .emph_current_underline AND .emph_current_bold
339      0465      2      THEN
340      0466      3      fs_wchar (fra, .outopt_und_char)
341      0467      2      ELSE
342      0468      2      fs_wchar (fra, %C' ');
343      0469      2
344      0470      2      [pass_real_text] :
345      0471      2      fs_wchar (fra, .character);
346      0472      2
347      0473      2      TES;
348      0474      2
349      0475      1      END;                                     ! End of OPEMPH
```


	54	00000000G	EF	9E	00002	MOVAB	OUTOPT, R4		
	53	00000000G	EF	9E	00009	MOVAB	FRA+4, R3		
	52	0C	AC	D0	00010	MOVL	ADR_EMPHASIS_BITS, R2	0410	
	51	1C	AC	D0	00014	MOVL	PASS_CNTR, RT	0412	
			3C	15	00018	BLEQ	6\$	0415	
	02		51	D1	0001A	CMPL	R1, #2		
			37	14	0001D	BGTR	6\$		
1A	62		02	E1	0001F	BBC	#2, (R2), 2\$	0419	
	16	04	A4	E9	00023	BLBC	OUTOPT+4, 2\$	0420	
	12	08	A4	E8	00027	BLBS	OUTOPT+8, 2\$	0421	
	01		51	D1	0002B	CMPL	R1, #1	0423	
			04	13	0002E	BEQL	1\$		
09	62		01	E1	00030	BBC	#1, (R2), 2\$		
	00	B3	64	90	00034	1\$: MOVB	OUTOPT, @FRA+4	0425	
			63	D6	00038	INCL	FRA+4		
		08	A3	D6	0003A	INCL	FRA+12		
	01		51	D1	0003D	2\$: CMPL	R1, #1	0428	
			04	13	00040	BEQL	3\$		
07	62		01	E1	00042	BBC	#1, (R2), 4\$		
	00	B3	04	AC	90	00046	3\$: MOVB	CHARACTER, @FRA+4	0430
			04	11	0004B	BRB	5\$		
	00	B3	20	90	0004D	4\$: MOVB	#32, @FRA+4	0432	
			63	D6	00051	5\$: INCL	FRA+4		
		08	A3	D6	00053	INCL	FRA+12	0430	
	05		51	D1	00056	6\$: CMPL	R1, #5	0435	
			29	19	00059	BLSS	11\$		
	06		51	D1	0005B	CMPL	R1, #6		
			24	14	0005E	BGTR	11\$		
	05		51	D1	00060	CMPL	R1, #5	0448	
			04	13	00063	BEQL	7\$		
12	62		01	E1	00065	BBC	#1, (R2), 9\$		
	50	14	AC	D0	00069	7\$: MOVL	OVERSTRIKE_CHAR, R0	0450	
	20		50	D1	0006D	CMPL	R0, #32		
			03	18	00070	BGEQ	8\$		
	50		20	D0	00072	MOVL	#32, R0		
	00	B3	50	90	00075	8\$: MOVB	R0, @FRA+4		
			04	11	00079	BRB	10\$		
	00	B3	20	90	0007B	9\$: MOVB	#32, @FRA+4	0452	
			63	D6	0007F	10\$: INCL	FRA+4		
		08	A3	D6	00081	INCL	FRA+12	0450	
	03		51	D1	00084	11\$: CMPL	R1, #3	0454	
			13	12	00087	BNEQ	14\$		
06	62		02	E1	00089	BBC	#2, (R2), 12\$	0456	
	00	B3	64	90	0008D	MOVB	OUTOPT, @FRA+4	0458	
			04	11	00091	BRB	13\$		
	00	B3	20	90	00093	12\$: MOVB	#32, @FRA+4	0460	
			63	D6	00097	13\$: INCL	FRA+4		
		08	A3	D6	00099	INCL	FRA+12	0458	
	04		51	D1	0009C	14\$: CMPL	R1, #4	0462	
			17	12	0009F	BNEQ	17\$		
0A	62		02	E1	000A1	BBC	#2, (R2), 15\$	0464	
06	62		01	E1	000A5	BBC	#1, (R2), 15\$		
	00	B3	64	90	000A9	MOVB	OUTOPT, @FRA+4	0466	
			04	11	000AD	BRB	16\$		
	00	B3	20	90	000AF	15\$: MOVB	#32, @FRA+4	0468	
			63	D6	000B3	16\$: INCL	FRA+4		
		08	A3	D6	000B5	INCL	FRA+12	0466	

Line output (emphasis -- bolding and underlining)
OPEMPH -- do emphasis by overprinting

H 2
16-Sep-1984 00:49:27
14-Sep-1984 13:06:55

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]LOEMPH.BLI:1

Page 12
(5)

00	07		51	D1	000B8	17\$:	CPL	R1, #7
			0A	12	000BB		BNEQ	18\$
	B3	04	AC	90	000BD		MOVB	CHARACTER, @FRA+4
			63	D6	000C2		INCL	FRA+4
		08	A3	D6	000C4		INCL	FRA+12
				04	000C7	18\$:	RET	

[illegible]

```
; Routine Size: 200 bytes,    Routine Base: $CODE$ + 00BD
```



```
351 U 0476 1 %IF FLIP %THEN
352 U 0477 1 %SBTTL 'FLEMPH -- process emphasized character for FLIP output'
353 U 0478 1 GLOBAL ROUTINE FLEMPH
354 U 0479 1 ( character
355 U 0480 1 , italics
356 U 0481 1 , adr_emphasis_bits
357 U 0482 1 , pass_cntr
358 U 0483 1 ) : NOVALUE =
359 U 0484 1
360 U 0485 1
361 U 0486 1 ++
362 U 0487 1 FUNCTIONAL DESCRIPTION:
363 U 0488 1 FLEMPH processes emphasis -- bolding and underlining -- for
364 U 0489 1 FLIP output (VMS only), if the user said /DEC_INTERNAL:FLIP.
365 U 0490 1
366 U 0491 1 FORMAL PARAMETERS:
367 U 0492 1
368 U 0493 1 character is the current character to be output (emphasized).
369 U 0494 1 It is -1 if emphasis is to be turned off.
370 U 0495 1 italics Not used by this routine -- passed for conformance only.
371 U 0496 1 adr_emphasis_bits Address of a word containing information on current-
372 U 0497 1 character and previous-character bold and underline.
373 U 0498 1 pass_cntr Count of which pass is happening.
374 U 0499 1
375 U 0500 1 IMPLICIT INPUTS: None
376 U 0501 1
377 U 0502 1 IMPLICIT OUTPUTS: None
378 U 0503 1
379 U 0504 1 ROUTINE VALUE:
380 U 0505 1 COMPLETION CODES: None
381 U 0506 1
382 U 0507 1 SIDE EFFECTS: None
383 U 0508 1 --
384 U 0509 1
385 U 0510 1 BEGIN
386 U 0511 1
387 U 0512 1 BIND emphasis_bits = .adr_emphasis_bits;
388 U 0513 1
389 U 0514 1 IF .character EQL -1
390 U 0515 1 THEN
391 U 0516 1 ! Turn emphasis off and return.
392 U 0517 1 BEGIN
393 U 0518 1 IF .emph_previous_bold
394 U 0519 1 THEN
395 U 0520 1 BEGIN
396 U 0521 1 emph_previous_bold = false;
397 U 0522 1 fs_wchar (fra, flip$k_end_bold);
398 U 0523 1 END;
399 U 0524 1 IF .emph_previous_underline
400 U 0525 1 THEN
401 U 0526 1 BEGIN
402 U 0527 1 emph_previous_underline = false;
403 U 0528 1 fs_wchar (fra, flip$k_end_underline);
404 U 0529 1 END;
405 U 0530 1 RETURN;
406 U 0531 1 END;
407 U 0532 1
```



```
: 408      U 0533 1      ! This call is a request to turn ON emphasis.
: 409      U 0534 1      SELECTONE 1 OF
: 410      U 0535 1      SET
: 411      U 0536 1
: 412      U 0537 1      [.emph_previous_bold AND (NOT .emph_current_bold)]:
: 413      U 0538 1      BEGIN
: 414      U 0539 1      fs_wchar (fra, flip$k_end_bold);
: 415      U 0540 1      emph_previous_bold = false;
: 416      U 0541 1      END;
: 417      U 0542 1
: 418      U 0543 1      [(NOT .emph_previous_bold) AND .emph_current_bold]:
: 419      U 0544 1      BEGIN
: 420      U 0545 1      fs_wchar (fra, flip$k_start_bold);
: 421      U 0546 1      emph_previous_bold = true;
: 422      U 0547 1      END;
: 423      U 0548 1
: 424      U 0549 1      [otherwise]:
: 425      U 0550 1      0;
: 426      U 0551 1
: 427      U 0552 1      TES;
: 428      U 0553 1
: 429      U 0554 1      SELECTONE 1 OF
: 430      U 0555 1      SET
: 431      U 0556 1
: 432      U 0557 1      [.emph_previous_underline AND (NOT .emph_current_underline)]:
: 433      U 0558 1      BEGIN
: 434      U 0559 1      fs_wchar (fra, flip$k_end_underline);
: 435      U 0560 1      emph_previous_underline = false;
: 436      U 0561 1      END;
: 437      U 0562 1
: 438      U 0563 1      [(NOT .emph_previous_underline) AND .emph_current_underline]:
: 439      U 0564 1      BEGIN
: 440      U 0565 1      fs_wchar (fra, flip$k_start_underline);
: 441      U 0566 1      emph_previous_underline = true;
: 442      U 0567 1      END;
: 443      U 0568 1
: 444      U 0569 1      [otherwise]:
: 445      U 0570 1      0;
: 446      U 0571 1
: 447      U 0572 1      TES;
: 448      U 0573 1
: 449      U 0574 1      !Now output the character to which the emphasis applies
: 450      U 0575 1      fs_wchar (fra, .character);
: 451      U 0576 1
: 452      U 0577 1      END;
: 453      U 0578 1      ! End of FLEMPH
```

%FI


```
455 U 0579 1 %IF DSRPLUS %THEN
456 U 0580 1 %SBTTL 'VTEMPH -- process emphasized character for VT100 output'
457 U 0581 1 GLOBAL ROUTINE VTEMPH
458 U 0582 1 ( character
459 U 0583 1 , italics
460 U 0584 1 , adr_emphasis_bits
461 U 0585 1 , pass_cntr
462 U 0586 1 ) : NOVALUE =
463 U 0587 1
464 U 0588 1
465 U 0589 1 FUNCTIONAL DESCRIPTION:
466 U 0590 1
467 U 0591 1 VTEMPH writes the proper escape sequences on the FRA to handle
468 U 0592 1 underlining and bolding if the user said /DEC_INTERNAL:VT100.
469 U 0593 1
470 U 0594 1 FORMAL PARAMETERS:
471 U 0595 1
472 U 0596 1 character is the current character to be output (emphasized).
473 U 0597 1 italics Not used by this routine -- passed for conformance only.
474 U 0598 1 It is -1 if emphasis is to be turned off.
475 U 0599 1 adr_emphasis_bits Address of a word containing information on current-
476 U 0600 1 character and previous-character bold and underline.
477 U 0601 1 pass_cntr Count of which pass is happening.
478 U 0602 1
479 U 0603 1 IMPLICIT INPUTS: None
480 U 0604 1
481 U 0605 1 IMPLICIT OUTPUTS: None
482 U 0606 1
483 U 0607 1 ROUTINE VALUE:
484 U 0608 1 COMPLETION CODES: None
485 U 0609 1
486 U 0610 1 SIDE EFFECTS: None
487 U 0611 1 --
488 U 0612 1
489 U 0613 1 BEGIN
490 U 0614 1
491 U 0615 1 BIND emphasis_bits = .adr_emphasis_bits;
492 U 0616 1
493 U 0617 1 $STR_DESC_INIT (DESCRIPTOR= work_string, CLASS=DYNAMIC);
494 U 0618 1
495 U 0619 1 ! Before processing the character, make sure that (1) there will be sufficient space in the
496 U 0620 1 ! output buffer to turn off emphasis and then turn it back on again, and (2) the output line
497 U 0621 1 ! is not too long for the operating system to handle.
498 U 0622 1 IF .fs_length (fra) GTR (.fs_maxsize (fra) - 20) ! Need at least 20 characters.
499 U 0623 1 OR
500 U 0624 1 .fs_length (fra) GTR max_output_line_length ! Reasonable output-line maximum length
501 U 0625 1 THEN
502 U 0626 1 BEGIN
503 U 0627 1 clh (clh_out_nocrlf);
504 U 0628 1 fs_init (fra);
505 U 0629 1 END;
506 U 0630 1
507 U 0631 1 IF .character EQL -1
508 U 0632 1 THEN
509 U 0633 1 ! Turn off all emphasis and return.
510 U 0634 1 BEGIN
511 U 0635 1 $STR_COPY (TARGET= work_string, STRING=
```



```
512 U 0636 1 $STR_CONCAT (%CHAR(escape), '[', '0', 'm' )
513 U 0637 1 );
514 U 0638 1 write_emphasis;
515 U 0639 1
516 U 0640 1 emph_previous_bold = false;
517 U 0641 1 emph_previous_underline = false;
518 U 0642 1
519 U 0643 1 RETURN;
520 U 0644 1 END;
521 U 0645 1
522 U 0646 1 ! This call is a request to turn ON emphasis.
523 U 0647 1
524 U 0648 1 ! If bolding and underlining are the same as for the previous character,
525 U 0649 1 ! do nothing. Otherwise, if there is a difference, we have to turn off
526 U 0650 1 ! everything so we can turn on only the one we want.
527 U 0651 1 IF (.emph_current_bold NEQ .emph_previous_bold)
528 U 0652 1 OR
529 U 0653 1 (.emph_current_underline NEQ .emph_previous_underline)
530 U 0654 1 THEN
531 U 0655 1 !Different status for at least one of them.
532 U 0656 1 BEGIN
533 U 0657 1 IF .emph_previous_emphasized NEQ 0
534 U 0658 1 THEN
535 U 0659 1 BEGIN
536 U 0660 1 ! Disable both of them (unless we're starting from a condition
537 U 0661 1 ! of no emphasis already).
538 U 0662 1 $STR_COPY (TARGET= work_string, STRING=
539 U 0663 1 $STR_CONCAT (%CHAR(escape), '[', '0', 'm' )
540 U 0664 1 );
541 U 0665 1 write_emphasis;
542 U 0666 1 END;
543 U 0667 1
544 U 0668 1 ! Turn off both history bits. The right ones will get turned on soon.
545 U 0669 1 emph_previous_bold = false;
546 U 0670 1 emph_previous_underline = false;
547 U 0671 1
548 U 0672 1 !Now turn on only the emphasis that's wanted.
549 U 0673 1 IF .emph_current_underline !Underlining wanted?
550 U 0674 1 THEN
551 U 0675 1 !Turn on underlining.
552 U 0676 1 BEGIN
553 U 0677 1 $STR_COPY (TARGET= work_string, STRING=
554 U 0678 1 $STR_CONCAT (%CHAR(escape), '[', '4', 'm' )
555 U 0679 1 );
556 U 0680 1 write_emphasis;
557 U 0681 1
558 U 0682 1 !Turn on the history bit for underlining.
559 U 0683 1 emph_previous_underline = true;
560 U 0684 1 END;
561 U 0685 1 IF .emph_current_bold !Bolding wanted?
562 U 0686 1 THEN
563 U 0687 1 !Turn on bolding.
564 U 0688 1 BEGIN
565 U 0689 1 $STR_COPY (TARGET= work_string, STRING=
566 U 0690 1 $STR_CONCAT (%CHAR(escape), '[', '1', 'm' )
567 U 0691 1 );
568 U 0692 1 write_emphasis;
```


LOEMPH
V04-000

Line output (emphasis -- bolding and underlining)
OPEMPH -- do emphasis by overprinting

M 2
16-Sep-1984 00:49:27
14-Sep-1984 13:06:55

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]LOEMPH.BLI;1

Page 17
(7)

```
: 569      U 0693 1
: 570      U 0694 1      !Turn on the history bit for bolding.
: 571      U 0695 1      emph_previous_bold = true;
: 572      U 0696 1      END;
: 573      U 0697 1      END;                                !End: turn on appropriate bits
: 574      U 0698 1
: 575      U 0699 1      !Now output the character to which the emphasis applies.
: 576      U 0700 1      fs_wchar (fra, .character);
: 577      U 0701 1
: 578      U 0702 1      END;                                ! End of VTEMPH
: 579      U 0703 1 %FI
```



```
581 0704 1 %IF LN01 %THEN
582 0705 1 %SBTTL 'LNEMPH -- process emphasized character for LN01 output'
583 0706 1 GLOBAL ROUTINE LNEMPH
584 0707 1 ( character
585 0708 1 , italics
586 0709 1 , adr_emphasis_bits
587 0710 1 , overstrike_count
588 0711 1 , overstrike_char
589 0712 1 , overstrike_seq
590 0713 1 , pass_cntr
591 0714 1 ) : NOVALUE =
592 0715 1
593 0716 1 ++
594 0717 1 FUNCTIONAL DESCRIPTION:
595 0718 1
596 0719 1 LNEMPH writes the proper escape sequences on the FRA to handle
597 0720 1 underlining and bolding if the user said /DEVECE=LN01[e].
598 0721 1
599 0722 1 The escape sequences written have the form of a font change:
600 0723 1
601 0724 1 <esc> [ N m
602 0725 1
603 0726 1 where N = 12 (text), 13 (bold), 14 (italic), or 15 (bold italic).
604 0727 1
605 0728 1 The SGR (select graphic rendition) escape sequences used for
606 0729 1 underlining have the same format exactly. N = 24 turns underlining
607 0730 1 on, and N = 4 turns it off.
608 0731 1
609 0732 1 FORMAL PARAMETERS:
610 0733 1
611 0734 1 character is the current character to be output (emphasized).
612 0735 1 It is -1 if emphasis is to be turned off.
613 0736 1 italics is TRUE if real italics (fonts 14 and 15) are to be
614 0737 1 used, and FALSE if underlining is to be used.
615 0738 1 adr_emphasis_bits Address of a word containing information on current-
616 0739 1 character and previous-character bold and underline.
617 0740 1 overstrike_count Number of characters in an overstrike sequence.
618 0741 1 overstrike_char The character with which to overstrike the previous one.
619 0742 1 overstrike_seq Not used by this routine -- passed for conformance only.
620 0743 1 pass_cntr Count of which pass is happening.
621 0744 1
622 0745 1 IMPLICIT INPUTS: None
623 0746 1
624 0747 1 IMPLICIT OUTPUTS: None
625 0748 1
626 0749 1 ROUTINE VALUE:
627 0750 1 COMPLETION CODES: None
628 0751 1
629 0752 1 SIDE EFFECTS: None
630 0753 1 --
631 0754 1
632 0755 2 BEGIN
633 0756 2
634 0757 2 BIND emphasis_bits = .adr_emphasis_bits;
635 0758 2
636 0759 2 LITERAL
637 0760 2 text_font = 12,
```



```

638 0761 2      bold_font = 13;
639 0762 2      italic_font = 14;
640 0763 2      bold_italic_font = 15;
641 0764 2
642 0765 2      LOCAL
643 0766 2          previous_font,
644 0767 2          new_font,
645 0768 2          underline_sgr;
646 0769 2
647 0770 2      ! Initialize.
648 0771 2
649 0772 2      previous_font = 0;
650 0773 2      new_font = 0;
651 0774 2      underline_sgr = 0;
652 0775 2      $STR_DESC_INIT (DESCRIPTOR= work_string, CLASS=DYNAMIC);
653 0776 2
654 0777 2      !**debug
655 0778 2      IF (.pass_cntr EQL pass_overstrike) AND .tsf_ovr
656 0779 2      THEN
657 0780 3          BEGIN
658 0781 4              IF (.overstrike_count NEQ 0)
659 0782 3              THEN
660 0783 4                  fs_wchar (fra, MAX(%C' ', .overstrike_char) )
661 0784 3              ELSE
662 0785 3                  fs_wchar (fra, %C' ');
663 0786 3              RETURN;
664 0787 2          END;
665 0788 2      !**end-debug
666 0789 2
667 0790 2      ! Before processing the character, make sure that (1) there will be
668 0791 2      ! sufficient space in the output buffer to turn off emphasis and then
669 0792 2      ! turn it back on again, and (2) the output line is not too long for
670 0793 2      ! the operating system to handle.
671 0794 2      IF .fs_length (fra) GTR (.fs_maxsize (fra) - 20)      ! Need at least 20 characters.
672 0795 2      OR
673 0796 2      .fs_length (fra) GTR max_output_line_length      ! Reasonable output-line maximum length
674 0797 2      THEN
675 0798 3          BEGIN
676 0799 3              clh (clh_out_nocrlf);
677 0800 3              fs_init (fra);
678 0801 3          END;
679 0802 2
680 0803 2      ! Calculate the font number used for the previous character. (It will be
681 0804 2      ! used to determine whether any font change is needed.) The calculation
682 0805 2      ! depends on the definition of emph_previous_emphasized (a macro), which
683 0806 2      ! is a 2-bit field having values 0, 1, 2, or 3. This value is added to the
684 0807 2      ! 'base' font (text_font = 12). If italics are not being used, the font
685 0808 2      ! number must be decremented by 2.
686 0809 2
687 0810 2      previous_font = .emph_previous_emphasized + text_font;
688 0811 2
689 0812 2      IF (.previous_font GEQ italic_font) AND (NOT .italics)
690 0813 2      THEN
691 0814 3          previous_font = .previous_font - 2;
692 0815 2
693 0816 2      IF .character EQL -1
694 0817 2      THEN
```



```
695 0818 2      ! Turn off all emphasis and return.
696 0819 BEGIN
697 0820      ! Reset to text font if not already in that font.
698 0821
699 0822      IF (.previous_font NEQ text_font)
700 0823      THEN
701 P 0824          $STR_COPY (TARGET= work_string, STRING=
702 P 0825              $STR_CONCAT ( %CHAR(escape)
703 P 0826                  , $STR_ASCII (text_font)
704 P 0827                  , 'm'
705 P 0828              );
706 P 0829
707 0830      );
708 0831
709 0832      ! Turn off underlining too if it was on.
710 0833
711 0834      IF (.emph_previous_underline      !Underlining on?
712 0835          AND NOT .italics)           !Really underlining, not italics?
713 0836      THEN
714 P 0837          $STR_APPEND (TARGET= work_string, STRING=
715 P 0838              $STR_CONCAT ( %CHAR(escape)
716 P 0839                  , '[24m'
717 P 0840              );
718 0841
719 0842      write_emphasis;
720 0843
721 0844      emph_previous_bold = false;
722 0845      emph_previous_underline = false;
723 0846
724 0847      RETURN;
725 0848      END;
726 0849
727 0850      + This call is a request to turn on emphasis. Determine what the new font
728 0851      should be. The calculation depends on the definition of emph_current_emphasized
729 0852      (a macro), which is a 2-bit field having values 0, 1, 2, or 3. This value is
730 0853      added to the 'base' font (text_font = 12). If italics are not being used
731 0854      the font number must be decremented by 2.
732 0855      -
733 0856      new_font = text_font + .emph_current_emphasized;
734 0857
735 0858      IF (.new_font GEQ italic_font) AND (NOT .italics)
736 0859      THEN
737 0860          new_font = .new_font - 2;
738 0861
739 0862      ! If we are doing underlining (not true italics), then set up the new
740 0863      font number to take advantage of the fact that the Underline On and
741 0864      Underline Off SGR escape sequences have exactly the same format as a
742 0865      font change.
743 0866
744 0867      IF NOT .italics
745 0868      THEN
746 0869          BEGIN
747 0870
748 0871          IF (.emph_current_underline AND (NOT .emph_previous_underline) )
749 0872          THEN
750 0873              underline_sgr = 4;          !Underline On SGR: <esc> [ 4 m
751 0874
```



```

752 0875 4      IF ( (NOT .emph_current_underline) AND .emph_previous_underline )
753 0876      THEN
754 0877          underline_sgr = 24;          !Underline Off SGR: <esc> [ 24 m
755 0878      END;
756 0879
757 0880      IF (.previous_font NEQ .new_font)
758 0881      OR (.underline_sgr NEQ 0)
759 0882      THEN
760 0883          BEGIN
761 0884              $STR_DESC_INIT (DESCRIPTOR= work_string, CLASS=DYNAMIC);
762 0885
763 0886              !Switch to the new font.
764 0887
765 0888              IF .previous_font NEQ .new_font
766 0889              THEN
767 0890                  $STR_COPY (TARGET= work_string, STRING=
768 0891                      $STR_CONCAT ( %CHAR(escape)
769 0892                          , $STR_ASCII (.new_font)
770 0893                          , 'm'
771 0894                      );
772 0895
773 0896              IF .underline_sgr NEQ 0
774 0897              THEN
775 0898                  $STR_APPEND (TARGET= work_string, STRING=
776 0899                      $STR_CONCAT ( %CHAR(escape)
777 0900                          , $STR_ASCII (.underline_sgr)
778 0901                          , 'm'
779 0902                      );
780 0903
781 0904              write_emphasis;
782 0905
783 0906              emph_previous_bold = .emph_current_bold;
784 0907              emph_previous_underline = .emph_current_underline;
785 0908
786 0909      END;
787 0910
788 0911      !Now output the character to which the emphasis applies
789 0912      fs_wchar (fra, .character);
790 0913
791 0914      END;
792 0915
793 0916      ! End of LNEMPH
794 0917
795 0918
796 0919 1

```

.PSECT \$PLITS,NOWRT,NOEXE,2

```

1B 00000 P.AAD: .ASCII <27>
5B 00001 P.AAE: .ASCII \[
6D 00002 P.AAF: .ASCII \m\
1B 00003 P.AAL: .ASCII <27>
6D 34 32 5B 00004 P.AAM: .ASCII \[24m\
1B 00008 P.AAS: .ASCII <27>
5B 00009 P.AAT: .ASCII \[

```



```

6D 0000A P.AAU: .ASCII \m\
1B 0000B P.ABB: .ASCII <27>
5B 0000C P.ABC: .ASCII \[
6D 0000D P.ABD: .ASCII \m\

.PSECT $OWNS$,NOEXE,2

0001 00008 $STR$STRING0:
      .WORD 1
01 0E 0000A .BYTE 14, 1
00000000' 0000C .ADDRESS P.AAD
0001 00010 $STR$STRING1:
      .WORD 1
01 0E 00012 .BYTE 14, 1
00000000' 00014 .ADDRESS P.AAE
0001 00018 $STR$STRING3:
      .WORD 1
01 0E 0001A .BYTE 14, 1
00000000' 0001C .ADDRESS P.AAF
0001 00020 $STR$STRING0:
      .WORD 1
01 0E 00022 .BYTE 14, 1
00000000' 00024 .ADDRESS P.AAL
0004 00028 $STR$STRING1:
      .WORD 4
01 0E 0002A .BYTE 14, 1
00000000' 0002C .ADDRESS P.AAM
0001 00030 $STR$STRING0:
      .WORD 1
01 0E 00032 .BYTE 14, 1
00000000' 00034 .ADDRESS P.AAS
0001 00038 $STR$STRING1:
      .WORD 1
01 0E 0003A .BYTE 14, 1
00000000' 0003C .ADDRESS P.AAT
0001 00040 $STR$STRING3:
      .WORD 1
01 0E 00042 .BYTE 14, 1
00000000' 00044 .ADDRESS P.AAU
0001 00048 $STR$STRING0:
      .WORD 1
01 0E 0004A .BYTE 14, 1
00000000' 0004C .ADDRESS P.ABB
0001 00050 $STR$STRING1:
      .WORD 1
01 0E 00052 .BYTE 14, 1
00000000' 00054 .ADDRESS P.ABC
0001 00058 $STR$STRING3:
      .WORD 1
01 0E 0005A .BYTE 14, 1
00000000' 0005C .ADDRESS P.ABD

$STR$DESC= WORK_STRING
$STR$BIN_DESC= WORK_STRING
$STR$TARGET= WORK_STRING
$STR$DESC= WORK_STRING
$STR$BIN_DESC= WORK_STRING

```


			50	DD	000C2	PUSHL	R0		
		10	A7	9F	000C4	PUSHAB	\$STR\$STRING1		
		08	A7	9F	000C7	PUSHAB	\$STR\$STRING0		
	69		04	FB	000CA	CALLS	#4, XST\$JOIN		
			5A	DD	000CD	PUSHL	R10		
			7E	D4	000CF	CLRL	-(SP)		
		0081	8F	BB	000D1	PUSHR	#^M<R0,R7>		
			7E	D4	000D5	CLRL	-(SP)		
1E	00000000G	EF	05	FB	000D7	CALLS	#5, XST\$COPY		
	62		04	E1	000DE	BBC	#4, (R2), 9\$		0834
	1A		08	AC	E8	BLBS	ITALICS, 9\$		0835
			28	A7	9F	PUSHAB	\$STR\$STRING1		0841
			20	A7	9F	PUSHAB	\$STR\$STRING0		
		69	02	FB	000EC	CALLS	#2, XST\$JOIN		
			5A	DD	000EF	PUSHL	R10		
			7E	D4	000F1	CLRL	-(SP)		
		0081	8F	BB	000F3	PUSHR	#^M<R0,R7>		
			7E	D4	000F7	CLRL	-(SP)		
	00000000G	EF	05	FB	000F9	CALLS	#5, XST\$APPEND		
	56		04	A7	D0	MOVL	WORK_STRING+4, PTR		
	51		67	3C	00104	MOVZWL	WORK_STRING, R1		
			50	D4	00107	CLRL	I		
			09	11	00109	BRB	11\$		
	00	B8	86	90	0010B	MOVB	(PTR)+, @FRA+4		
			68	D6	0010F	INCL	FRA+4		
			08	A8	D6	INCL	FRA+12		
F3		50	51	F3	00114	AOBLEQ	R1, I, 10\$		
		62	18	8A	00118	BICB2	#24, (R2)		0845
				04	0011B	RET			0819
54	62	02	01	EF	0011C	EXTZV	#1, #2, (R2), NEW_FONT		0856
		54	0C	C0	00121	ADDL2	#12, NEW_FONT		
		0E	54	D1	00124	CMPL	NEW_FONT, #14		0858
			07	19	00127	BLSS	13\$		
		1D	08	AC	E8	BLBS	ITALICS, 16\$		
		54	02	C2	0012D	SUBL2	#2, NEW_FONT		0860
		16	08	AC	E8	BLBS	ITALICS, 16\$		0867
08		62	02	E1	00134	BBC	#2, (R2), 15\$		0871
03		62	04	E0	00138	BBS	#4, (R2), 14\$		
		53	04	D0	0013C	MOVL	#4, UNDERLINE_SGR		0873
		62	02	E0	0013F	BBS	#2, (R2), 16\$		0875
07		62	04	E1	00143	BBC	#4, (R2), 16\$		
03		53	18	D0	00147	MOVL	#24, UNDERLINE_SGR		0877
			50	D4	0014A	CLRL	R0		0880
		54	55	D1	0014C	CMPL	PREVIOUS_FONT, NEW_FONT		
			04	13	0014F	BEQL	17\$		
			50	D6	00151	INCL	R0		
			07	11	00153	BRB	18\$		
			53	D5	00155	TSTL	UNDERLINE_SGR		0881
			03	12	00157	BNEQ	18\$		
			0093	31	00159	BRW	23\$		
	67	020E0000	8F	D0	0015C	MOVL	#34471936, \$STR\$DESC		0885
		04	A7	D4	00163	CLRL	\$STR\$DESC+4		
	2B		50	E9	00166	BLBC	R0, 19\$		0890
			7E	D4	00169	CLRL	-(SP)		0898
			54	DD	0016B	PUSHL	NEW_FONT		
	7E		8F	3C	0016D	MOVZWL	#2307, -(SP)		
	6B		03	FB	00172	CALLS	#3, XST\$ASCII		

		40	A7	9F	00175	PUSHAB	\$STR\$STRING3		
			50	DD	00178	PUSHL	R0		
		38	A7	9F	0017A	PUSHAB	\$STR\$STRING1		
		30	A7	9F	0017D	PUSHAB	\$STR\$STRING0		
	69		04	FB	00180	CALLS	#4, XST\$JOIN		
			5A	DD	00183	PUSHL	R10		
			7E	D4	00185	CLRL	-(SP)		
		0081	8F	BB	00187	PUSHR	#^M<R0,R7>		
			7E	D4	0018B	CLRL	-(SP)		
	00000000G	EF	05	FB	0018D	CALLS	#5, XST\$COPY		
			53	D5	00194	TSTL	UNDERLINE_SGR		0900
			2B	13	00196	BEQL	20\$		
			7E	D4	00198	CLRL	-(SP)		0908
			53	DD	0019A	PUSHL	UNDERLINE_SGR		
	7E	0903	8F	3C	0019C	MOVZWL	#2307, -(SP)		
	6B		03	FB	001A1	CALLS	#3, XST\$ASCII		
		58	A7	9F	001A4	PUSHAB	\$STR\$STRING3		
			50	DD	001A7	PUSHL	R0		
		50	A7	9F	001A9	PUSHAB	\$STR\$STRING1		
		48	A7	9F	001AC	PUSHAB	\$STR\$STRING0		
	69		04	FB	001AF	CALLS	#4, XST\$JOIN		
			5A	DD	001B2	PUSHL	R10		
			7E	D4	001B4	CLRL	-(SP)		
		0081	8F	BB	001B6	PUSHR	#^M<R0,R7>		
			7E	D4	001BA	CLRL	-(SP)		
	00000000G	EF	05	FB	001BC	CALLS	#5, XST\$APPEND		
		53	A7	D0	001C3	MOVL	WORK_STRING+4, PTR		
		51	67	3C	001C7	MOVZWL	WORK_STRING, R1		
			50	D4	001CA	CLRL	I		
			09	11	001CC	BRB	22\$		
	00	B8	83	90	001CE	MOVB	(PTR)+, @FRA+4		
			68	D6	001D2	INCL	FRA+4		
		08	A8	D6	001D4	INCL	FRA+12		
			51	F3	001D7	AOBLEQ	R1, I, 21\$		
50	F3	50	01	EF	001DB	EXTZV	#1, #1, (R2), R0		0911
62	01	01	50	F0	001E0	INSV	R0, #3, #1, (R2)		
50	62	01	02	EF	001E5	EXTZV	#2, #1, (R2), R0		0912
62	01	04	50	F0	001EA	INSV	R0, #4, #1, (R2)		
		00	B8	AC	90	001EF	MOVB	CHARACTER, @FRA+4	0917
				68	D6	001F4	INCL	FRA+4	
			08	A8	D6	001F6	INCL	FRA+12	
				04	001F9	RET			0919

; Routine Size: 506 bytes, Routine Base: \$CODE\$ + 0185

: 797 0920 1 %FI
: 798 0921 1
: 799 0922 1 END
: 800 0923 0 ELUDOM

! End of module

PSECT SUMMARY

Name	Bytes	Attributes						
\$OWNS	96	NOVEC,	WRT,	RD	NOEXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)
\$CODE\$	895	NOVEC,NOWRT,	RD	EXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)	
\$SPLITS	14	NOVEC,NOWRT,	RD	NOEXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)	

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]XPORT.L32;1	590	51	8	252	00:00.1
_\$255\$DUA28:[RUNOFF.SRC]DSRLIB.L32;1	1248	36	2	86	00:00.3

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:LOEMPH/OBJ=OBJ\$:LOEMPH MSRC\$:LOEMPH/UPDATE=(ENH\$:LOEMPH)

Size:

895 code + 110 data bytes

Run Time:

00:46.6

Elapsed Time:

01:37.1

Lines/CPU Min:

1188

Lexemes/CPU-Min:

88197

Memory Used:

294 pages

Compilation Complete

0343

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY